# Sampling period dependent RST controller used in control/scheduling co-design

David Robert, Olivier Sename, Daniel Simon

# SAMPLING PERIOD DEPENDENT RST CONTROLLER USED IN CONTROL/SCHEDULING CO-DESIGN

**David Robert** * **Olivier Sename** * **Daniel Simon** **

* *Laboratoire d'automatique de Grenoble, ENSIEG, BP 46 38402 Saint Martin d'Heres Cedex, France*

** *INRIA Rhone-Alpes, 655 avenue de l'Europe, Montbonnot, 38334 Saint-Ismier Cedex, France*

Abstract: The paper presents a sampling period dependent RST controller used in the context of control/scheduling co-design. This plant controller is implemented on a real-time operating system as a control task. A scheduling controller adapts control task periods to regulate the processor load. The link established between scheduling and plant control provides more flexibility and robustness w.r.t. the variation of the processor load. Parameterized RST and scheduling controller synthesis are presented and a co-design example illustrates the approach flexibility.


Keywords: Computer controlled systems, digital control, RST controller, feedback scheduling

## 1. INTRODUCTION

Digital control systems are often implemented as a set of tasks running on top off a real-time operating system. Control tasks are generally viewed by the scheduling community as hard real-time tasks with fixed sampling periods and known "worst case execution time" (WCET). On the other hand control community supposes that periods are constants and can not be changed on-line. Theses assumptions are often too restrictive. For instance many control laws can tolerate small variations on the sampling period without leading to instability. On the other hand WCET based scheduling of control laws with varying execution time leads to an under utilization of the processor unit. Moreover, practical estimation of the WCET is a difficult and time consuming work. Finally embedded applications with complex control laws require flexibility to allocate computational resources on the fly.

In this paper a feedback scheduler regulates the processor utilization to avoid overload. It acts on the task periods and measures task loads. Plant control laws, computed by the tasks, should fit the sampling period variations, therefore a sampling period dependant RST controller is proposed. The link established between scheduling and plant control provides more flexibility and robustness w.r.t. the variation of the processor load.

Control and scheduling co-design is a recent interest in both the computer and control communities. A first approach uses off-line co-design. In (Seto *et al.*, 1996) optimal periods which maximize the control law performances w.r.t. the resource constraints are obtained by solving an optimization problem. In (Ryu and Hong, 1998)

control law performances depend on the periods as well as on input/output latencies and an heuristic algorithm computes the optimal periods. In (Palopoli *et al.*, 2002) an optimization problem is solved to find optimal task periods and feedback gains. The previous approaches do not bring flexibility therefore on-line adaptation was studied. In (Cervin and Eker, 2000) a feedback controller with a sampling period dependant PID controller is used. In (Cervin *et al.*, 2002; Eker *et al.*, 2000) rescaling factors obtained by off-line optimization preserve the optimality of a set of control task periods. In (Caccamo *et al.*, 2000; Sha *et al.*, 2000) a modified Constant Bandwidth Server adapts task periods to locally handle an overrun when the execution time varies. In (Cervin and Eker, 2003) a framework, based on a Constant Bandwidth Server, is used to enhance the determinism of the co-design. In (Sename *et al.*, 2003) the authors have developed a LQ controller to regulate the processor load and a delay dependant robust controller to stabilize the plant w.r.t. latencies. In the computer community, Wei and Yu (2003) and Lu *et al.* (2002) use two PID controllers to regulate the deadline miss-ratio and the CPU consumption of real-time tasks.

The outline of this paper is as follow. Section 2 describes the scheduling model and its controller design. Section 3 presents a parameterized polynomial pole placement synthesis used for the plant control. In the section 4 constant and variable plant controller performances are compared. An illustrative example of the co-design is presented in section 5. Finally, the paper ends with some conclusions and further research directions.

## 2. SCHEDULING CONTROLLER

Plant control task periods are on-line adjusted according to the processor load variations. This work is done by a specific task, the scheduling controller, which has the highest priority and a period much larger than the others tasks.

Figure (1) presents the bloc diagram of the feedback scheduling. The scheduling is viewed as a dynamical system which output is the processor utilization and inputs are control task frequencies. As far as the adaptation of the control tasks is concerned, the load of the other tasks is seen as an output disturbance.

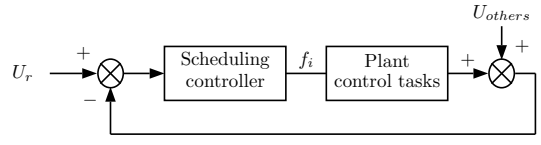


Fig. 1. Feedback scheduling bloc diagram

The scheduling is here limited to periodic tasks. In this case the processor load induced by a task is defined by $U = \frac{c}{h}$ where $c$ and $h$ are the execution time and period of the task. It can be rewritten as $U = c\ f$, where $f$ is the task frequency, which is linear when the execution time is constant.

Based on this equation, processor load induced by a task is modelled in a way similar to Cervin *et al.* (2002). Thus, for each period $h_s$ of the scheduling controller, the processor load of one plant control task is estimated as :

$$\hat{U}_{kh_s} = \lambda\, \hat{U}_{(k-1)h_s} + (1-\lambda)\, \overline{c}_{kh_s}\, f_{(k-1)h_s} \quad (1)$$

where $f$ is the sampling frequency currently assigned to the plant control task and $\overline{c}$ is the mean of its measured job execution-time. $\lambda$ is a forgetting factor used to smooth the measure.

As $\overline{c}$ depends on the runtime environment (e.g. processor speed) a "normalized" linear model of the task $i$ (2) is used for the scheduling controller synthesis where $\overline{c}$ is omitted and will be compensated by on-line gain-scheduling $(1/\overline{c})$, as illustrated by the figure (4) in section 5.

$$G_i(z) = \frac{\hat{U}_i(z)}{f_i(z)} = \frac{1-\lambda_i}{z-\lambda_i} \quad (2)$$

Finally, the scheduling of $n$ plant control tasks is modelled by the discrete-time state-space representation :

$$\begin{cases} x(k+1) &= Ax(k) + Bf(k) \\ \hat{u}(k) &= Cx(k) \end{cases} \quad (3)$$

with $A = diag\{\lambda_1, \ldots, \lambda_n\}$, $B = diag\{1-\lambda_1, \ldots, 1-\lambda_n\}$ and $C = [1 \ldots 1]$. $f$ and $x$ are respectively the vectors of task frequencies and task loads whereas $\hat{u}$ is the load of all plant control tasks.

Based on this state-space representation standard control methodology can be used to design a controller. Here a discrete $H_\infty$ synthesis is proposed and illustrated in section 5.

## 3. PLANT CONTROL DESIGN

As the scheduling controller adapts the periods of the plant control tasks, the latter should fit the new sampling period $h$ in order to preserve stability. The design objective is to obtain an unique controller as a function of $h$ instead of a map of different controllers. Thus the stability can be theoretically ensured for all $h$. A polynomial pole-placement approach is used as explained below.

The structure of the plant controller, in figure (1), is a well-known two degrees of freedom discrete-

time controller. The desired closed-loop performances are specified by model matching, as done in (Åström and Wittenmark, 1997).

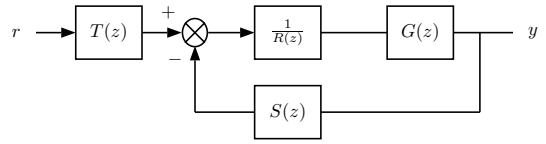


Fig. 2. Plant controller bloc diagram

As $h$ is not constant, it is easier to specify the closed-loop model in a continuous-time form. Adding a dependance on $h$ allows it to be parameterized by the sampling period (as emphasized in section 4). The plant and closed-loop models are expressed by the transfer functions :

$$G(s) = \frac{B(s)}{A(s)} \text{ and } G_m(s,h) = \frac{B_m(s,h)}{A_m(s,h)} \quad (4)$$

A formal discretization, exact or approximative, with $h$ as parameter, leads to both discrete-time transfer functions $G(z,h)$ and $G_m(z,h)$ as below, with $i \leq j$ :

$$G(z,h) = \frac{B(z,h)}{A(z,h)} = \frac{b_i(h)z^i + ... + b_0(h)}{z^j + a_{j-1}(h)z^{j-1} + ... + a_0(h)} \quad (5)$$

For easier reading the dependence in the $z$ and $h$ variables will sometimes be missed out. Then, the closed-loop controlled system of figure (2) can be expressed by :

$$G_{cl}(z,h) = \frac{B(z,h)\,T(z,h)}{A(z,h)\,R(z,h) + B(z,h)\,S(z,h)} \quad (6)$$

The synthesis objective is to find $R(z,h)$, $S(z,h)$ and $T(z,h)$ such as $G_{cl}(z,h)$ matches $G_m(z,h)$. In a way similar to Åström and Wittenmark (1997) the following factorizations (7) are used, where $A$ and $B$ do not have any common factors, $A^+$ and $B^+$ are monic and contained stable poles and zeros which can be cancelled, $A^-$ and $B^-$ contained unstable poles and zeros, $R_d$ and $S_d$ specify given factors of $R$ and $S$ (e.g. integral terms).

$$\begin{array}{rlrl} A &= A^+A^- & S &= A^+S_dS' \\ B &= B^+B^- & R &= B^+R_dR' \\ B_m &= B^-B_m' & T &= A^+T' \end{array} \quad (7)$$

The matching problem (8) needs to solve the Diophantine equation (9). $R$, $S$ and $T$ are obtained with (7). $A_o$ is the observer polynomial which will be defined, see (Åström and Wittenmark, 1997).

$$\frac{B^+A^+B^-T'}{B^+A^+(A^-R_dR' + B^-S_dS')} = \frac{B^-B_m'}{A_m} \quad (8)$$

$$(A^-R_dR' + B^-S_dS') = A_mA_o \quad (9)$$

$$T' = B_m'A_o \quad (10)$$

Necessary and sufficient conditions to obtain an unique solution and a causal controller, when computation time is disregarded, are given on polynomial degrees in $z$, by : for all $h \in \mathbb{R}^+$,

$$d^oA_m - d^oB_m = d^oA - d^oB \quad (11)$$

$$d^oR = d^oS = d^oT \quad (12)$$

$$d^oA_o = d^oA + d^oS_d + d^oR_d + d^oA^- - d^oB^+ - 1 - d^oA_m \quad (13)$$

The internal observer should be faster than the closed-loop, therefore the roots of $A_o$ are chosen in accordance with $A_m$ and (13). The observer dynamic can be defined as a function of the closed-loop dynamic (e.g. five times faster) or arbitrary defined by a continuous-time model discretized as for the closed-loop model.

*Remark 1.* Due to the dependence in $h$, equation (9) has to be solved analytically, which can be done with any symbolic computation software.

*Remark 2.* $h$-dependent plant model of equation (5) could also be the result of an interpolation of identifications issue at different sampling periods.

*Remark 3.* Conditions (11-13) should be generically satisfy, i.e. for almost all values of $h$. In practice the designer can select a set of $h$ for which none of polynomials in (11-13) loses a degree in $z$.

## 4. CONSTANT VS VARIABLE PLANT CONTROLLER PERFORMANCES

As introduced in section 3 the plant controller can be designed with constant or sampling period dependent performance specifications. Both approaches are here compared in an example.

### *4.1 Plant controller synthesis*

The plant is a stable pendulum. The continuous model (14), linearized at the equilibrium, expressed the angular position in function of the applied torque with $\omega_0 = 3.77\ rad/s$, $\xi = 0.2$ and $K = \omega_0/9.81$, see (Eker *et al.*, 2000).

$$G(s) = \frac{K}{s^2 + 2\xi\omega_0 s + \omega_0^2} \quad (14)$$

Closed-loop performances are defined in continuous-time form by (15) with $\xi_m = 0.7$ and $K_m = \omega_m^2$ for a unity static gain. Internal observer $A_o$ is chosen five times faster than the closed-loop model therefore $\omega_{obs} = 5\ \omega_m$ in (16). A zero steady state error to a step disturbance is required.

$$G_m(s,h) = \frac{K_m(h)}{s^2 + 2\xi_m\omega_m(h)s + \omega_m^2(h)} \quad (15)$$

$$A_o(s,h) = s^2 + 2\xi_m\omega_{obs}(h)s + \omega_{obs}^2(h) \quad (16)$$

Both cases are considered :

- Constant closed-loop performances $\omega_m = 10\ rad/s$
- Variable closed-loop performances
  According to the rule of thumb of Åström and Wittenmark (1997), $\omega_{obs}\ h \approx 0.2 \ldots 0.6$ because $\omega_{obs}$ is the highest pulsation of the closed-loop system. By choosing the middle of the interval, $\omega_m = 1/5\ \omega_{obs} = 4/(50\ h)$.

Then $G(s)$ and $G_m(s,h)$ are discretized with Tustin's approximation. Two discrete-time transfer functions are obtained as :

$$G(z,h) = \frac{K_d(h)\ (z+1)^2}{z^2 + a_{1d}(h)\ z + a_{0d}(h)} \quad (17)$$

By defining $B^- = (z+1)^2$, which appears in both $G$ and $G_m$ numerators, $B^+ = 1$, $A^+ = 1$, $S_d = 1$ and $R_d = (z-1)$ for the integral action, equation (9) and (10) are solved. Using conditions (12) and (13) we obtain $degT = degS = degR = 2$, $degR' = 1$ and $degA_0 = 2$. Solving the Diophantine equation (9) leads to the three polynomials (18) where each parameters are expressed by a fourth order rational function in $h$.

$$R(z,h) = (z-1)\ (r_1(h)\ z + r_0(h)) \quad (18)$$

$$S(z,h) = s_2(h)\ z^2 + s_1(h)\ z + s_0(h) \quad (19)$$

$$T(z,h) = t_2(h)\ z^2 + t_1(h)\ z + t_0(h) \quad (20)$$

### *4.2 Simulation*

The two plant control methods designed in the previous section are now compared for the sampling periods 2, 8, 15, 25, and 50 $ms$ using a non-linear model of the pendulum.

Figure (3,a) presents the step response when closed-loop performance specifications are constant. The system becomes unstable for sampling periods greater than 25 $ms$. According to the rule of thumb $\omega_{obs}\ h \approx 0.2 \ldots 0.6$, a good sampling period for this case is $h \in [4; 12]\ ms$. It appears that this rule is here more restrictive than necessary which is due to a minimum phase margin guaranteed by the rule.

Figure (3,b) presents the step response when closed-loop performances depend on the sampling period. As it was predictable, the system is never unstable and the performances (i.e. $\omega_m$) decrease when the sampling period increases.

To conclude this section, adapting closed-loop performance specifications according to the sampling period can increase robust stability w.r.t. sampling period changes. The next section presents the benefits of adapting closed-loop performances in the context of control/scheduling co-design.

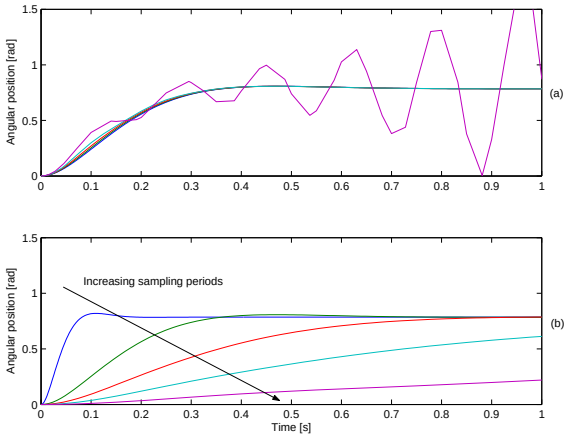


Fig. 3. Step response of the closed-loop with constant (a) or variable (b) performance specifications for the sampling periods 2 to 50ms

## 5. CO-DESIGN EXAMPLE

In this example, two independent stable pendulums are controlled by a computer. Four tasks share the same processor unit on top of a real-time operating system with priority based scheduling. Two tasks control the both pendulums, one task implements the scheduling controller and the last is a disruptive task. Tables (1) and (2) summarize scheduling and pendulum properties respectively. Pendulum control laws are designed as explained in the previous section.

Table 1. Scheduling properties

| Task | Priority | Period (ms) | Execution time(ms) |
|---|---|---|---|
| schedctrl | 1 | 500 | 1 |
| pend1 | 2 | 4 to 400 | 2 |
| disturb | 3 | 4 | 2 |
| pend2 | 4 | 4 to 400 | 2 |

Table 2. Plant properties

| Pendulum | Task | $\xi$ | $\omega_o$ | $\xi_m$ | $\omega_m$ |
|---|---|---|---|---|---|
| 1 | pend1 | 0.2 | 3.77 | 0.7 | 10 |
| 2 | pend2 | 0.2 | 4.08 | 0.7 | 10 |

### *5.1 Scheduler control design*

The scheduling controller has to adjust the two plant control task periods when the desired processor utilization varies, in response to a reference change or a disruptive task.

Figure (4) presents the bloc diagram of this example which includes saturations and on-line gain-scheduling used to compensate the variations of the job execution time, see section 2.

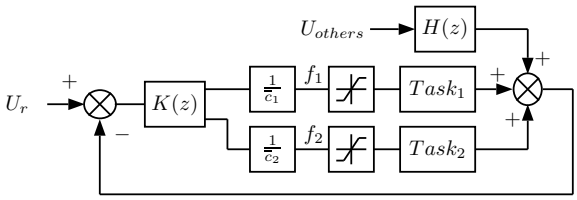


Fig. 4. Control scheme for CPU resources

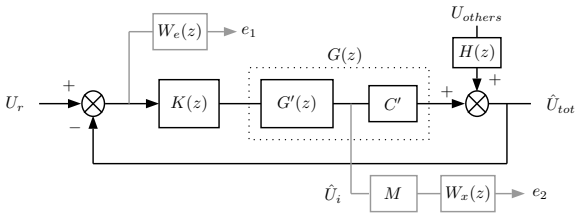


Fig. 5. $H_\infty$ design bloc diagram

The bloc diagram of figure (5) is considered for the $H_\infty$ design where $G'(z)$ is defined by (3) with $C = I_2$ to obtain a full state output and $C' = [1\ 1]$. $H(z)$ is the dynamic of the sensor which measures the load of the other tasks. It can be a first order filter as in (1) but not necessarily. The template $W_e$ specifies the performances on the load tracking error as follows :

$$W_e(s) = \frac{s/M_s + \omega_b}{s + \omega_s \epsilon} \tag{21}$$

with $M_s = 2$, $\omega_s = 1\ rad/s$, $\epsilon = 0.05$ to obtain a closed-loop rise time of 3 $s$, a static error less than 5 % and a good robustness margin. Matrix $M = [-\alpha\ 1]$ and template $W_x$ allow to specify the load allocation between the two control tasks. With a large gain in $W_x$, $U_2 - \alpha\, U_1 \approx 0$ i.e. $\frac{U_2}{U_1} \approx \alpha$.

All templates are discretized with a sampling period of 500 $ms$. Finally discrete-time $H_\infty$ synthesis, using the LMI Control Toolbox of Matlab, produces a discrete-time scheduling controller of order 3.

### *5.2 Co-simulation*

This co-simulation is done using a non-linear model of the pendulum and the Truetime Matlab Toolbox, see (Cervin *et al.*, 2003). A specific Simulink bloc simulates a real-time kernel with tasks, scheduler and input/output capabilities used to connect it with classical Simulink dynamic system blocs.

Two examples are presented, the first with two pendulums with constant closed-loop performances and the second with variable ones.

The scheduling scenario is the same for both examples. Processor utilization starts with a reference of 80 %. At time $t = 5\ s$ reference decreases to 60 %, representing a decrease of available resources. At time $t = 12\ s$ a new (disruptive) task appears which needs 50 % of the processor resources. The priorities and the load allocation ratio $\alpha = 2$ are chosen to favor the first pendulum. The two pendulums are excited by square wave reference signal of period 6 $s$.

### *5.3 Example 1, constant closed-loop performances*

In this example, the two control laws have constant closed-loop performance specifications summarized in the table 2. A good sampling period for these performances is $h \in [4; 12]\ ms$, see section 4.

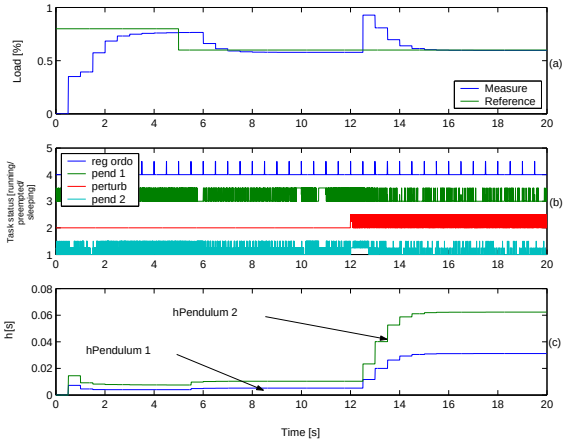


Fig. 6. Example 1, scheduling : (a) processor load, (b) scheduling timing, (c) control task periods

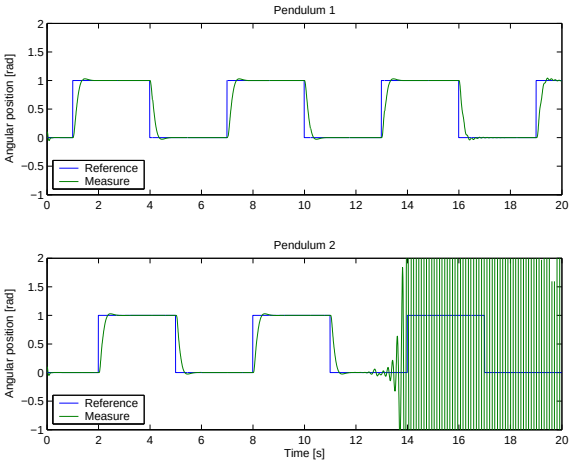


Fig. 7. Example 1, pendulum positions

In figure (6), the periods of the two pendulum control laws are adjusted when the processor load reference changes ($t = 5\ s$) or when load disturbance appears ($t = 12\ s$). Due to the load allocation ratio, the two periods are not identical. As the execution time is the same for the two control tasks (in this example) the allocation ratio can be seen on the periods, for example at $t = 18\ s$ $\frac{h_2}{h1} \approx 2 = \alpha$.

In figure (7), both pendulums remain stable after the load reference change. The pendulum two becomes unstable when the disruptive task appears because the sampling period becomes too high.

### *5.4 Example 2, varying closed-loop performances*

In this example, pendulum 1 (high priority) has constant performances and pendulum 2 (low priority) has varying performances, see section 4. The scheduling behavior is the same as the first example, therefore it is not shown here.

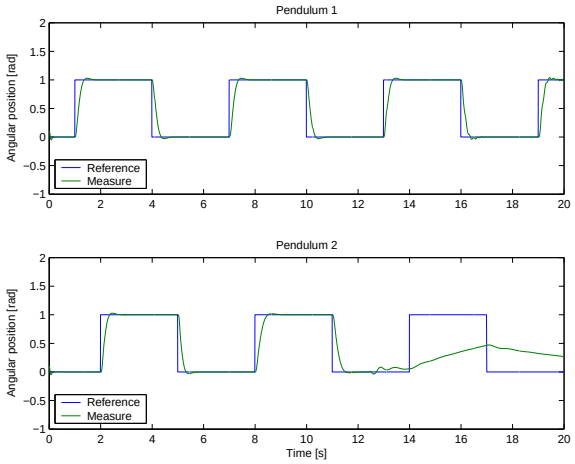


Fig. 8. Example 2, pendulum positions

In figure (8) both pendulums stay stable even if the sampling periods rise sharply at $t = 12\ s$. The stability of the pendulum 2 is preserved by decreasing the performances. This example emphasizes the interest for adapting closed-loop performances w.r.t. sampling period.

## 6. CONCLUSION AND FUTURE WORK

In this paper a processor load regulation has been presented based on a simple scheduling model and $H_\infty$ synthesis. The synthesis of a sampling period dependant RST controller for the plant has been exposed. Then it has been shown that a co-design, with these two controllers and varying plant performances, can improve robustness in stability and flexibility w.r.t. processor load variations.

However this work is only at its first stage. The scheduling model use a measure of execution time which may be difficult to get in practice, especially with off-the-self operating systems. In this case an observer should be design to estimate the execution time. On the other hand the RST synthesis procedure is complex to used with higher order plant due to the high order, in $h$, of the R, S and T polynomial parameters. The next studies will use parameter dependent controller synthesis such as LPV.

## REFERENCES


Åström, K.J. and B. Wittenmark (1997). *Computer-Controlled Systems*. Information and systems sciences series. 3rd ed.. Prentice Hall. New Jersey.

Caccamo, M., G. Buttazzo and L. Sha (2000). Elastic feedback control. In: *Proc. 12th Euromicro Conference on Real-Time Systems*. Stockholm, Sweden.

Cervin, A. and J. Eker (2000). Feedback scheduling of control tasks. In: *Proc. 39th IEEE Conference on Decision and Control*. Sydney, Australia.

Cervin, A. and J. Eker (2003). The Control Server: A computational model for real-time control tasks. In: *Proc. 15th Euromicro Conference on Real-Time Systems*. Porto, Portugal.

Cervin, A., D. Henriksson, B. Lincoln, J. Eker and K.E. Årzén (2003). How does control timing affect performance?. *IEEE Control Systems Magazine* **23**(3), 16–30.

Cervin, A., J. Eker, B. Bernhardsson and K.E. Årzén (2002). Feedback-feedforward scheduling of control tasks. *Real-Time Systems* **23**(1–2), 25–53.

Eker, J., P. Hagander and K.E. Årzén (2000). A feedback scheduler for real-time controller tasks. *Control Engineering Practice* **8**(12), 1369–1378.

Lu, C., J.A. Stankovic, S.H. Son and G. Tao (2002). Feedback control real-time scheduling: Framework, modeling and algorithms. *Real-Time Systems Journal* **23**(1/2), 85–126.

Palopoli, L., C. Pinello, A. Sangiovanni-Vincentelli, L. Elghaoui and A. Bicchi (2002). Synthesis of robust control systems under resource constraints. In: *Hybrid Systems: Computation and Control*. Standford.

Ryu, M. and S. Hong (1998). Toward automatic synthesis of schedulable real-time controllers. *International Journal of Integrated Computer-Aided Engineering* **5**(3), 261–277.

Sename, O., D. Simon and D. Robert (2003). Feedback scheduling for real-time control of systems with communication delays. In: *Proc. IEEE Confer. on Emerging Technologies and Factory Automation*. Lisbon, Portugal.

Seto, D., J.P. Lehoczky, L. Sha and K.G. Shin (1996). On task schedulability in real-time control systems. In: *Proc. 17th IEEE Real-Time Systems Symposium*. Washington, DC.

Sha, L., X. Liu, M. Caccamo and G. Buttazzo (2000). Online control optimization using load driven scheduling. In: *Proc. 39th IEEE Conference on Decision and Control*. Sydney, Australia.

Wei, L. and H. Yu (2003). Research on a soft real-time scheduling algorithm based on hybrid adaptive control architecture. In: *Proc. 2003 American Control Conference*.